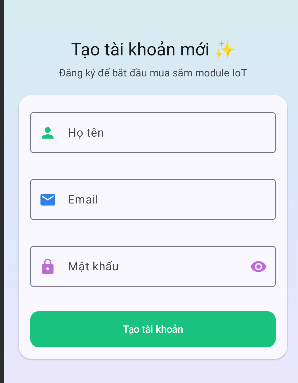
LAB 8   
Trương Gia Huy 1150080019

ảnh giao diện   
A screenshot of a phone

AI-generated content may be incorrect.

code của giao diện   
package com.example.buymodule.ui  
  
import androidx.compose.foundation.background  
import androidx.compose.foundation.layout.\*

import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*Email*import androidx.compose.material.icons.filled.*Lock*import androidx.compose.material.icons.filled.*Visibility*import androidx.compose.material.icons.filled.*VisibilityOff*import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.graphics.Brush  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.text.input.PasswordVisualTransformation  
import androidx.compose.ui.text.input.VisualTransformation  
import androidx.compose.ui.unit.dp  
import com.example.buymodule.vm.AuthVm  
import kotlinx.coroutines.launch  
  
@Composable  
fun LoginScreen(  
 authVm: AuthVm,  
 onLoggedIn: () -> Unit,  
 onOpenRegister: () -> Unit  
) {  
 // State  
 var email by remember **{** *mutableStateOf*("") **}** var pass by remember **{** *mutableStateOf*("") **}** var showPwd by remember **{** *mutableStateOf*(false) **}** var busy by remember **{** *mutableStateOf*(false) **}** var emailErr by remember **{** *mutableStateOf*<String?>(null) **}** var passErr by remember **{** *mutableStateOf*<String?>(null) **}** val scope = rememberCoroutineScope()  
 val snackbar = remember **{** SnackbarHostState() **}** val token by authVm.token.collectAsState()  
 val error by authVm.error.collectAsState()  
  
 LaunchedEffect(token) **{** if (!token.*isNullOrBlank*()) onLoggedIn()  
 busy = false  
 **}** LaunchedEffect(error) **{** if (error != null) {  
 snackbar.showSnackbar(error!!)  
 busy = false  
 }  
 **}** // Gradient nền rực rỡ hơn  
 val rainbow = Brush.verticalGradient(  
 *listOf*(  
 *Color*(0xFF19C37D).copy(.18f), // xanh ngọc  
 *Color*(0xFF2F80ED).copy(.14f), // xanh dương  
 *Color*(0xFFBB6BD9).copy(.12f) // tím  
 )  
 )  
  
 Scaffold(snackbarHost = **{** SnackbarHost(snackbar) **}**) **{** p **->** Box(  
 modifier = Modifier  
 .*padding*(p)  
 .*fillMaxSize*()  
 .*background*(rainbow),  
 contentAlignment = Alignment.Center // ✅ đặt form vào giữa màn hình  
 ) **{** // Container để giới hạn bề ngang cho đẹp  
 Column(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(horizontal = 20.*dp*)  
 .*widthIn*(max = 420.*dp*),  
 horizontalAlignment = Alignment.CenterHorizontally  
 ) **{** // Header gradient chip  
 Box(  
 modifier = Modifier  
 .*background*(  
 brush = Brush.horizontalGradient(  
 *listOf*(*Color*(0xFF19C37D), *Color*(0xFF2F80ED))  
 ),  
 shape = *RoundedCornerShape*(16.*dp*)  
 )  
 .*padding*(horizontal = 16.*dp*, vertical = 8.*dp*)  
 ) **{** Text("Chào mừng trở lại 👋", color = Color.White)  
 **}** Spacer(Modifier.*height*(8.*dp*))  
 Text(  
 "Đăng nhập để tiếp tục mua sắm module IoT",  
 style = MaterialTheme.typography.bodyMedium,  
 color = MaterialTheme.colorScheme.onSurfaceVariant  
 )  
  
 Spacer(Modifier.*height*(20.*dp*))  
  
 // Thẻ form  
 ElevatedCard(  
 modifier = Modifier.*fillMaxWidth*(),  
 shape = *RoundedCornerShape*(18.*dp*),  
 colors = CardDefaults.elevatedCardColors(  
 containerColor = MaterialTheme.colorScheme.surface  
 )  
 ) **{** Column(Modifier.*padding*(16.*dp*)) **{** OutlinedTextField(  
 value = email,  
 onValueChange = **{** email = **it** emailErr = null  
 **}**,  
 label = **{** Text("Email") **}**,  
 leadingIcon = **{** Icon(Icons.Filled.*Email*, null, tint = *Color*(0xFF2F80ED)) **}**,  
 singleLine = true,  
 isError = emailErr != null,  
 supportingText = **{** emailErr?.*let* **{** Text(**it**) **} }**,  
 modifier = Modifier.*fillMaxWidth*(),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = *Color*(0xFF2F80ED),  
 focusedLabelColor = *Color*(0xFF2F80ED),  
 cursorColor = *Color*(0xFF2F80ED)  
 )  
 )  
  
 Spacer(Modifier.*height*(12.*dp*))  
  
 OutlinedTextField(  
 value = pass,  
 onValueChange = **{** pass = **it** passErr = null  
 **}**,  
 label = **{** Text("Mật khẩu") **}**,  
 leadingIcon = **{** Icon(Icons.Filled.*Lock*, null, tint = *Color*(0xFFBB6BD9)) **}**,  
 trailingIcon = **{** IconButton(onClick = **{** showPwd = !showPwd **}**) **{** Icon(  
 if (showPwd) Icons.Filled.*VisibilityOff* else Icons.Filled.*Visibility*,  
 contentDescription = null,  
 tint = *Color*(0xFFBB6BD9)  
 )  
 **}  
 }**,  
 singleLine = true,  
 isError = passErr != null,  
 supportingText = **{** when {  
 passErr != null -> Text(passErr!!)  
 pass.*isNotEmpty*() && pass.length < 6 ->  
 Text("Tối thiểu 6 ký tự")  
 }  
 **}**,  
 visualTransformation = if (showPwd) VisualTransformation.None  
 else PasswordVisualTransformation(),  
 modifier = Modifier.*fillMaxWidth*(),  
 colors = OutlinedTextFieldDefaults.colors(  
 focusedBorderColor = *Color*(0xFFBB6BD9),  
 focusedLabelColor = *Color*(0xFFBB6BD9),  
 cursorColor = *Color*(0xFFBB6BD9)  
 )  
 )  
  
 Spacer(Modifier.*height*(18.*dp*))  
  
 Button(  
 onClick = **{** var ok = true  
 if (!email.*contains*("@") || email.length < 5) {  
 emailErr = "Email không hợp lệ"  
 ok = false  
 }  
 if (pass.length < 6) {  
 passErr = "Mật khẩu quá ngắn"  
 ok = false  
 }  
 if (!ok) return@Button  
 scope.*launch* **{** busy = true  
 authVm.login(email.*trim*(), pass)  
 **}  
 }**,  
 enabled = token.*isNullOrBlank*() && !busy,  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(50.*dp*),  
 shape = *RoundedCornerShape*(12.*dp*),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = *Color*(0xFF2F80ED)  
 )  
 ) **{** if (busy) {  
 CircularProgressIndicator(  
 strokeWidth = 2.*dp*,  
 color = Color.White,  
 modifier = Modifier.*size*(18.*dp*)  
 )  
 Spacer(Modifier.*width*(10.*dp*))  
 }  
 Text("Đăng nhập", color = Color.White)  
 **}** TextButton(  
 onClick = onOpenRegister,  
 modifier = Modifier.*align*(Alignment.End)  
 ) **{** Text(  
 "Tạo tài khoản",  
 color = *Color*(0xFF19C37D)  
 )  
 **}  
 }  
 }  
 }  
 }  
 }**}

A screenshot of a phone

AI-generated content may be incorrect.

@file:OptIn(ExperimentalMaterial3Api::class)  
  
package com.example.buymodule.ui  
  
import androidx.compose.foundation.background  
import androidx.compose.foundation.clickable  
import androidx.compose.foundation.layout.\*  
import androidx.compose.foundation.rememberScrollState  
import androidx.compose.foundation.shape.*CircleShape*import androidx.compose.foundation.shape.RoundedCornerShape  
import androidx.compose.foundation.verticalScroll  
import androidx.compose.material.icons.Icons  
import androidx.compose.material.icons.filled.*ArrowBack*import androidx.compose.material.icons.rounded.*Star*import androidx.compose.material3.\*  
import androidx.compose.runtime.\*  
import androidx.compose.ui.Alignment  
import androidx.compose.ui.Modifier  
import androidx.compose.ui.draw.clip  
import androidx.compose.ui.draw.shadow  
import androidx.compose.ui.graphics.Brush  
import androidx.compose.ui.graphics.Color  
import androidx.compose.ui.layout.ContentScale  
import androidx.compose.ui.platform.*LocalContext*import androidx.compose.ui.text.font.FontWeight  
import androidx.compose.ui.text.style.TextDecoration  
import androidx.compose.ui.text.style.TextOverflow  
import androidx.compose.ui.unit.dp  
import coil.compose.AsyncImage  
import coil.request.ImageRequest  
import com.example.buymodule.data.dto.ModuleDto  
import com.example.buymodule.data.repo.Repo  
import com.example.buymodule.vm.CartVm  
import kotlinx.coroutines.launch  
import java.text.NumberFormat  
import java.util.Locale  
import kotlin.math.roundToInt  
  
@Composable  
fun ModuleDetailScreen(  
 id: String,  
 repo: Repo,  
 cartVm: CartVm,  
 onBack: () -> Unit,  
 onBuyNow: (() -> Unit)? = null  
) {  
  
 // Brand palette (khớp ShopHomeScreen)  
 val BrandGreen = *Color*(0xFF19C37D)  
 val BrandBlue = *Color*(0xFF2F80ED)  
 val BrandWhite = *Color*(0xFFFFFFFF)  
 val BrandGradient = Brush.horizontalGradient(*listOf*(BrandGreen, BrandBlue))  
  
 var item by remember **{** *mutableStateOf*<ModuleDto?>(null) **}** var loading by remember **{** *mutableStateOf*(true) **}** var error by remember **{** *mutableStateOf*<String?>(null) **}** val scope = rememberCoroutineScope()  
  
 LaunchedEffect(id) **{** try {  
 loading = true; error = null  
 item = repo.getModule(id)  
 } catch (e: Exception) {  
 error = e.message  
 } finally { loading = false }  
 **}** Scaffold(  
 topBar = **{** TopAppBar(  
 title = **{** Text(  
 item?.name ?: "Chi tiết",  
 maxLines = 1,  
 overflow = TextOverflow.Ellipsis  
 )  
 **}**,  
 navigationIcon = **{** IconButton(onClick = onBack) **{** Icon(Icons.Filled.*ArrowBack*, contentDescription = "Back")  
 **}  
 }** )  
 **}**,  
 bottomBar = **{** // Thanh action giống Shopee: Add to cart + Buy now  
 if (item != null) {  
 val m = item!!  
 val price = m.price ?: 0.0  
 val discount = 0 // nếu backend có discountPercent thì thay vào  
 val finalPrice = price  
  
 Surface(color = MaterialTheme.colorScheme.surface) **{** Row(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*padding*(12.*dp*),  
 verticalAlignment = Alignment.CenterVertically  
 ) **{** Column(Modifier.*weight*(1f)) **{** val nf = remember **{** NumberFormat.getCurrencyInstance(Locale.*US*) **}** Row(verticalAlignment = Alignment.Bottom) **{** Text(  
 nf.format(finalPrice),  
 style = MaterialTheme.typography.titleLarge.copy(fontWeight = FontWeight.Bold),  
 color = MaterialTheme.colorScheme.primary  
 )  
 if (discount > 0) {  
 Spacer(Modifier.*width*(6.*dp*))  
 Text(  
 nf.format(price),  
 style = MaterialTheme.typography.labelLarge.copy(  
 color = MaterialTheme.colorScheme.outline  
 ),  
 textDecoration = TextDecoration.LineThrough  
 )  
 }  
 **}  
 }** // Add to cart  
 Button(  
 onClick = **{** m.id?.*let* **{** scope.*launch* **{** cartVm.add(**it**) **} }  
 }**,  
 modifier = Modifier  
 .*weight*(1f)  
 .*clip*(*RoundedCornerShape*(12.*dp*))  
 .*background*(BrandGradient),  
 colors = ButtonDefaults.buttonColors(  
 containerColor = Color.Transparent,  
 contentColor = BrandWhite  
 )  
 ) **{** Text("Thêm vào giỏ", fontWeight = FontWeight.SemiBold) **}** Spacer(Modifier.*width*(10.*dp*))  
  
 // Buy now (tuỳ chọn điều hướng sau)  
 OutlinedButton(  
 onClick = **{** m.id?.*let* **{** scope.*launch* **{** cartVm.add(**it**)  
 onBuyNow?.invoke()**} }** // *TODO: điều hướng sang màn Checkout nếu có* **}**,  
 modifier = Modifier.*weight*(1f),  
 shape = *RoundedCornerShape*(12.*dp*),  
 colors = ButtonDefaults.outlinedButtonColors(  
 contentColor = BrandBlue  
 ),  
 border = ButtonDefaults.outlinedButtonBorder.copy(  
 brush = BrandGradient  
 )  
 ) **{** Text("Đặt mua") **}  
 }  
 }** }  
 **}** ) **{** padding **->** when {  
 loading -> Box(Modifier.*fillMaxSize*().*padding*(padding)) **{** CircularProgressIndicator(Modifier.*align*(Alignment.Center))  
 **}** error != null -> Box(Modifier.*fillMaxSize*().*padding*(padding)) **{** Text(error ?: "Lỗi", color = MaterialTheme.colorScheme.error, modifier = Modifier.*align*(Alignment.Center))  
 **}** item == null -> Box(Modifier.*fillMaxSize*().*padding*(padding)) **{** Text("Không tìm thấy sản phẩm", modifier = Modifier.*align*(Alignment.Center))  
 **}** else -> {  
 val m = item!!  
 val nf = remember **{** NumberFormat.getCurrencyInstance(Locale.*US*) **}** val price = m.price ?: 0.0  
 val discount = 0  
 val finalPrice = price  
  
 Column(  
 modifier = Modifier  
 .*fillMaxSize*()  
 .*verticalScroll*(rememberScrollState())  
 .*padding*(padding)  
 ) **{** // Ảnh lớn + overlay nhẹ  
 Box(  
 modifier = Modifier  
 .*fillMaxWidth*()  
 .*height*(260.*dp*)  
 .*background*(  
 Brush.verticalGradient(  
 *listOf*(BrandBlue.copy(0.06f), Color.Transparent)  
 )  
 )  
 ) **{** val imgUrl = m.imageUrl  
 ?: "https://picsum.photos/900?random=${m.id ?: m.name.hashCode()}"  
 AsyncImage(  
 model = ImageRequest.Builder(*LocalContext*.current)  
 .data(imgUrl)  
 .crossfade(true)  
 .build(),  
 contentDescription = m.name,  
 contentScale = ContentScale.Crop,  
 modifier = Modifier.*fillMaxSize*()  
 )  
  
 if (discount > 0) {  
 Box(  
 modifier = Modifier  
 .*align*(Alignment.TopStart)  
 .*padding*(12.*dp*)  
 .*clip*(*RoundedCornerShape*(8.*dp*))  
 .*background*(BrandGreen.copy(0.15f))  
 .*padding*(horizontal = 10.*dp*, vertical = 6.*dp*)  
 ) **{** Text("-$discount%", color = BrandGreen, fontWeight = FontWeight.SemiBold)  
 **}** }  
 **}** // Tên + rating + sold + kho  
 Column(Modifier.*padding*(16.*dp*)) **{** Text(  
 m.name,  
 style = MaterialTheme.typography.titleLarge,  
 fontWeight = FontWeight.SemiBold  
 )  
  
 Spacer(Modifier.*height*(6.*dp*))  
  
 Row(verticalAlignment = Alignment.CenterVertically) **{** val rating = 4.6f // nếu backend có m.rating thì thay vào  
 *repeat*(5) **{** i **->** val tint =  
 if (i < rating.*coerceIn*(0f, 5f).*roundToInt*()) BrandGreen  
 else MaterialTheme.colorScheme.outline  
 Icon(  
 Icons.Rounded.*Star*, null,  
 tint = tint,  
 modifier = Modifier.*size*(18.*dp*)  
 )  
 **}** Spacer(Modifier.*width*(8.*dp*))  
 Text(  
 "Đã bán ${m.description?.length ?: 0}",  
 style = MaterialTheme.typography.labelLarge,  
 color = MaterialTheme.colorScheme.onSurfaceVariant  
 )  
 Spacer(Modifier.*width*(8.*dp*))  
 Text(  
 "Tồn ${m.inStock ?: 0}",  
 style = MaterialTheme.typography.labelLarge,  
 color = MaterialTheme.colorScheme.onSurfaceVariant  
 )  
 **}** Spacer(Modifier.*height*(14.*dp*))  
  
 // Giá (hiển thị lại ở nội dung chính cho rõ)  
 Row(verticalAlignment = Alignment.Bottom) **{** Text(  
 nf.format(finalPrice),  
 style = MaterialTheme.typography.headlineSmall.copy(fontWeight = FontWeight.Bold),  
 color = MaterialTheme.colorScheme.primary  
 )  
 if (discount > 0) {  
 Spacer(Modifier.*width*(8.*dp*))  
 Text(  
 nf.format(price),  
 style = MaterialTheme.typography.titleSmall.copy(  
 color = MaterialTheme.colorScheme.outline  
 ),  
 textDecoration = TextDecoration.LineThrough  
 )  
 }  
 **}** Spacer(Modifier.*height*(18.*dp*))  
  
 // Mô tả  
 Text("Mô tả", style = MaterialTheme.typography.titleMedium, fontWeight = FontWeight.SemiBold)  
 Spacer(Modifier.*height*(6.*dp*))  
 Text(  
 m.description ?: "Không có mô tả.",  
 style = MaterialTheme.typography.bodyMedium,  
 color = MaterialTheme.colorScheme.onSurface  
 )  
  
 // Specs (nếu có)  
 m.category?.*let* **{** Spacer(Modifier.*height*(16.*dp*))  
 Row(  
 verticalAlignment = Alignment.CenterVertically,  
 modifier = Modifier  
 .*clip*(*RoundedCornerShape*(10.*dp*))  
 .*background*(BrandBlue.copy(0.06f))  
 .*padding*(horizontal = 12.*dp*, vertical = 8.*dp*)  
 ) **{** Text("Loại:", fontWeight = FontWeight.SemiBold)  
 Spacer(Modifier.*width*(6.*dp*))  
 Text(**it**)  
 **}  
 }** Spacer(Modifier.*height*(90.*dp*)) // chừa chỗ cho bottomBar  
 **}  
 }** }  
 }  
 **}**}

A screenshot of a phone

AI-generated content may be incorrect.